AMENDMENTS TO THE SPECIFICATION

Please replace paragraph [1001] with the following amended paragraph:

[1001] This application is related to (1) U.S. Patent Application No. 09/996,088 entitled "AGGRESSIVE PREFETCH OF ADDRESS CHAINS," naming Peter Damron and Nicolai Kosche as inventors, and filed 28 November 2001 and to (2) U.S. Patent Application No. 10/050,358~~xx/xxx,xxx [Att'y Dkt. No. 004-7047]~~ entitled " TECHNIQUE FOR ASSOCIATING INSTRUCTIONS WITH EXECUTION EVENTS," naming Nicolai Kosche, Brian J. Wylie, Christopher P. Aoki, and Peter C. Damron as inventors, filed on even date herewith. Each of the related applications is incorporated herein by reference in its entirety.

Please replace paragraph [1005] with the following amended paragraph:

[1005] One reasonably comprehensive hardware assisted profiling environment is provided by the Digital Continuous Profiling Infrastructure (DCPI) tools that run on Alpha processor systems to provide profile information at several levels of granularity, from whole images down to individual procedures and basic blocks on down to detailed information about individual instructions, including information about dynamic behavior such as cache misses, branch mispredicts and other forms of dynamic stalls. Detailed information on the DCPI tools and downloadable code may be found (at least as of the filing date) online ~~at http://www.research.digital.com/SRC/dcpi or at http://www.tru64unix.compaq.com/dcpi~~. Additional descriptive information appears in Jennifer Anderson, Lance Berc, George Chrysos, Jeffrey Dean, Sanjay Ghemawat, Jamey Hicks, Shun-Tak Leung, Mitch Lichtenberg, Mark Vandevoorde, Carl A. Waldspurger, William E. Weihl, "Transparent, Low-Overhead Profiling on Modern Processors," in *Proceedings of the Workshop on Profile and Feedback-Directed Compilation* in conjunction with the *International Conference on Parallel Architectures and Compilation Techniques* (*PACT 98*), Paris, France (October 13, 1998).

Please replace paragraph [1024] with the following amended paragraph:

[1024] While a variety of techniques may be employed to associate particular execution events with particular tag-identified instructions, some embodiments in accordance with the present invention employ techniques described in greater detail in the above-incorporated U.S. Patent Application No. 10/050,358~~xx/xxx,xxx [Att'y Dkt. No. 004-7047]~~ entitled "TECHNIQUE FOR ASSOCIATING INSTRUCTIONS WITH EXECUTION EVENTS," and naming Nicolai Kosche, Brian J. Wylie, Christopher P. Aoki and Peter C. Damron as inventors, filed on even date herewith. Such techniques address certain challenges associated with delayed event detection and ambiguity creating locations in an instruction sequence. By way of example, **FIG. 2B** illustrates an instruction sequence executable on a processor such as that illustrated in **FIG. 1**. The instruction sequence includes a load instruction **203** for which association of a detected cache miss execution event may be complicated by the presence of an ambiguity creating location. Absent the ambiguity creating location, backtracking from the miss detection to load instruction **203** is straightforward. However, in the illustrated case, interposed instruction **205** is a branch target of one or more branch or control transfer instructions (not specifically shown). Because detection of a cache miss corresponding to load instruction **203** is delayed (e.g., by detection lag **210**), and because the branch target location is interposed between the target load instruction and a point in the instruction sequence (namely, instruction **206**) that coincides with detection of the cache miss, execution path ambiguity complicates the association of the detected cache miss with load instruction **203**. Absent additional information, it is unclear whether the actual execution that caused the cache miss took a path that includes instructions **203, 204, 205, ... 206**, or whether actual execution arrived at instruction **205** via a branch or other control transfer. In the latter case, some memory access instruction other than load instruction **203** caused the detected cache miss and it should not be associated with instruction **203**.